Asian Journal of Basic Science & Research
Volume 6, Issue 2, Pages 145-158, April-June 2024

Predicting Stock Price by Using Attention-Based Hybrid LSTM Model

Jiawei Zhou[1*]

[1]*Khoury College of Computer Sciences, Northeastern University, Vancouver, Canada.*
Corresponding Author Email: zhou.jiaw@northeastern.edu[*]

DOI: http://doi.org/10.38177/AJBSR.2024.6211

Article Received: 08 April 2024 *Article Accepted: 15 June 2024* *Article Published: 21 June 2024*

ABSTRACT

The financial markets are inherently complex and dynamic, characterized by high volatility and the influence of numerous factors. Predicting stock price trends is a challenging endeavor that has been significantly advanced by the advent of machine learning techniques. This study investigates the effectiveness of various models, including Long Short-Term Memory (LSTM), XGBoost, Support Vector Machine (SVM), and hybrid models combining LSTM with XGBoost and SVM, in forecasting stock prices. Our results indicate that the LSTM model outperforms others, demonstrating superior predictive accuracy. Among the hybrid models, LSTM combined with XGBoost shows the best performance. Despite these findings, the study identifies several areas for further improvement, such as enhanced feature engineering, advanced hybrid models, refined attention mechanisms, improved model interpretability, data augmentation, and real-time prediction capabilities. This research contributes valuable insights into the application of hybrid attention-based LSTM models in financial forecasting, highlighting their potential and areas for future enhancement.

Keywords: Stock price prediction; Machine learning; LSTM; XGBoost; Attention mechanisms; Hybrid model; Financial forecasting; Predictive accuracy; Feature engineering.

1. Introduction

The financial markets are inherently complex and dynamic, characterized by their high volatility and the influence of numerous factors such as economic indicators, market sentiment, and geopolitical events. Predicting stock price trends has long been a challenging endeavor for traders and analysts aiming to maximize profits (Hu et al., 2015). In recent years, the advent of advanced machine learning techniques has opened new avenues for making more accurate and reliable predictions. Among these techniques, Long Short-Term Memory (LSTM) networks have shown significant promise due to their ability to capture temporal dependencies in sequential data.

Traditional time series models, such as the Autoregressive Integrated Moving Average (ARIMA) model (Asteriou & Hall, 2011) and the Generalized Auto-Regressive Conditional Heteroskedasticity (GARCH) model (Engle, 2001), have long been employed for stock forecasting. However, these models face significant limitations in accurately capturing nonlinear time series data, as they require several prerequisites that are often unmet in the complex and volatile financial markets. At the same time, sequence modeling remains a crucial area of research within artificial intelligence, impacting a variety of fields (Jin et al., 2021). Neural networks, known for their substantial ability to generalize nonlinear patterns, are the most commonly employed AI techniques. However, neural networks also come with their own set of challenges: they require large amounts of data to train effectively, are prone to overfitting, and can be computationally intensive (Srivastava, 2013). Additionally, the training process of neural networks can be difficult to interpret, leading to challenges in understanding and trusting the model outputs, which is particularly critical in financial applications (Khandelwal et al., 2021).

The Long Short-Term Memory (LSTM) model is a prominent choice among recurrent neural network (RNN) architectures for analyzing sequential data. Unlike traditional RNNs, LSTM networks incorporate memory cells designed to evaluate the relevance of information. Each memory cell includes three gates: the forget gate, the input

gate, and the output gate (Jin et al., 2021). When data enters an LSTM network, it is processed through these gates to determine its importance. Irrelevant information is discarded via the forget gate, while pertinent information is retained and used for future processing. This structure allows LSTMs to effectively capture long-term dependencies in financial time series data.

Attention mechanisms, on the other hand, provide a way to dynamically focus on different parts of the input sequence when making predictions. They work by assigning varying levels of importance, or "attention weights," to different input elements based on their relevance to the current task (Vaswani et al., 2017). This enables the model to concentrate on the most critical pieces of information at each step.

Incorporating attention mechanisms into LSTM networks, creating attention-based LSTM models, enhances their performance by allowing the network to prioritize significant information dynamically. This is especially beneficial in financial time series analysis, where certain data points may be more crucial than others for accurate fore-casting. By focusing on the most relevant data, attention-based LSTM models can achieve improved predictive accuracy and better handle the complexity and volatility inherent in financial markets.

In this study, attention-based LSTM models will be employed to investigate the optimal hybrid attention-based LSTM approach for stock price prediction. To accomplish this goal, attention-based LSTM+XGBoost and attention-based LSTM+SVM will be evaluated to ascertain the superior hybrid model for stock forecasting. This research marks the inaugural attempt to contrast these hybrid models within this domain.

Attention-based LSTM+XGBoost combines the strengths of both LSTM networks and XGBoost. LSTMs are effective at capturing temporal dependencies and patterns in sequential data, while XGBoost is a powerful gradient boosting algorithm known for its high performance and accuracy in handling structured data. This hybrid approach leverages LSTM's ability to model long-term dependencies and XGBoost's robustness in classification and regression tasks, potentially leading to more accurate and reliable stock price predictions.

Attention-based LSTM+SVM integrates LSTM networks with Support Vector Machines (SVM). LSTMs capture the sequential and temporal aspects of the data, while SVMs are effective for their high accuracy and generalization capabilities, particularly in high-dimensional spaces. This combination aims to enhance the model's ability to capture both complex temporal patterns and achieve precise classification or regression results, making it a strong candidate for stock price prediction.

By comparing various models, including hybrid attention-based LSTM models, this paper aims to identify which approach offers superior performance in the context of stock price prediction, providing valuable insights into the efficacy of hybrid attention-based LSTM models in financial forecasting.

1.1. Study Objectives

The primary objective of this study is to evaluate the effectiveness of hybrid attention-based LSTM models in predicting stock prices, specifically by comparing the performance of two hybrid approaches: attention-based LSTM+XGBoost and attention-based LSTM+SVM. This research aims to explore whether incorporating attention mechanisms into LSTM networks can enhance their ability to capture long-term dependencies and improve

prediction accuracy in the volatile financial market. Additionally, the study seeks to determine the relative strengths of integrating LSTMs with XGBoost, a gradient boosting algorithm known for its robust handling of structured data, versus combining LSTMs with Support Vector Machines (SVM), which are renowned for their accuracy in high-dimensional spaces. By conducting a comparative analysis of these hybrid models, the study intends to identify the optimal approach for stock forecasting, thereby offering valuable insights into the practical applications of advanced machine learning techniques in financial markets. Furthermore, the research will contribute to understanding how attention mechanisms can dynamically prioritize relevant information in financial time series data, potentially leading to more reliable and interpretable model outputs.

2. Literature Review

The fusion of statistical techniques with learning models has significantly advanced several machine learning algorithms, such as artificial neural networks, gradient-boosted regression trees, support vector machines, and random forests. These sophisticated algorithms excel at identifying intricate, non-linear patterns and relationships that are difficult to detect using linear models. Additionally, they exhibit superior performance and manage multicollinearity more effectively compared to linear regression methods. Numerous studies are currently focused on applying machine learning methods in finance. Some research has utilized tree-based models to predict portfolio returns (Moritz & Zimmermann, 2016), while others have employed deep learning to forecast future values of financial assets (Takeuchi & Lee, 2013). Additionally, various researchers have evaluated different methods to achieve accurate stock market predictions. These methods range from evolutionary computation using genetic algorithms (Allen & Karjalainen, 1999) to statistical learning with algorithms like Support Vector Machines (SVM) (Kurani et al., 2023).

Examining studies on deep learning in stock markets, Batres-Estrada (2015) shows that deep learning algorithms, particularly a Deep Belief Network (DBN) combined with a Multilayer Perceptron (MLP), surpass traditional methods in predicting financial data and selecting stock portfolios. These deep learning models deliver superior and more consistent returns compared to benchmarks like logistic regression, a standalone multilayer perceptron, and a naive benchmark, highlighting the potential of deep learning approaches in the finance sector. Jing et al. (2021) propose a hybrid model that integrates a Convolutional Neural Network for sentiment analysis with a Long Short-Term Memory Neural Network for technical indicator analysis, showing enhanced accuracy in predicting stock prices on the Shanghai Stock Exchange compared to standalone models and those lacking sentiment analysis. Aggarwal et al. (2023) present a model that integrates sentiment scores of financial news with an MLP-Regressor to forecast stock prices. Their study reveals that amalgamating sentiment scores from three algorithms yields high accuracy, reaching a peak of 0.90 for 10-day trend predictions. However, certain stocks, such as Tata Motors, display elevated prediction errors.

LSTM has collected significant attention for stock price prediction. Sen et al. (2021) introduce a hybrid modeling method for this purpose, employing both machine learning and deep learning techniques, notably LSTM networks, validated through walk-forward validation. Their findings highlight the success of LSTM-based univariate models in using one-week prior data to forecast NIFTY 50 open values. Wu et al. (2021) proposes a novel framework,

SACLSTM, combining Convolutional Neural Net- work (CNN) and Long-Short-Term Memory Neural Network (LSTM), to achieve more accurate stock price predictions by constructing a sequence array of historical data and leading indicators and extracting feature vectors through CNN before inputting the into LSTM, showing improved prediction performance compared to previous methods. Zhang (2022) invents a PCA-LSTM model for stock price prediction, which effectively reduces input data dimensionality using principle component analysis and incorporates stock-related technical indicators such as KDJ and MACD, resulting in improved prediction accuracy, reduced running time, and enhanced stability.

Furthermore, while many studies have explored the effectiveness of LSTM models in stock price prediction, there is a lack of comparative analysis regarding hybrid LSTM models. This research endeavors to bridge this gap by conducting a comprehensive comparison between two hybrid models: an attention-based SVM and LSTM hybrid model, and an attention-based XGBoost and LSTM hybrid model. Through this comparative analysis, insights into the relative performance and efficacy of these hybrid approaches can be gained, contributing to the advancement of predictive modeling in financial markets.

3. Methodology

In this section, we will discuss LSTM, XGBoost, SVM and Hybrid models.

3.1. LSTM

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network (RNN) designed to effectively model temporal sequences and handle long-range dependencies (Gers et al., 2021). LSTMs overcome the vanishing gradient problem, which is prevalent in traditional RNNs, through a sophisticated gating mechanism that regulates information flow.

Here is the LSTM Mechanism. An LSTM unit computes a sequence of hidden states h = (h_1, h_2, . . . , h_m) and cell states C = (C_1, C_2, . . . , C_m) as follows:

1. **Forget Gate:** The forget gate f_t determines how much of the previous cell state C_{t-1} should be forgotten. It is defined as:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \qquad (1)$$

where σ is the logistic sigmoid function, W_f is the weight matrix, and b_f is the bias vector for the forget gate.

2. **Input Gate:** The input gate i_t controls how much of the new information from the current input x_t and the previous hidden state h_{t-1} should be added to the cell state. It is given by:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \qquad (2)$$

where W_i and b_i are the weight matrix and bias vector for the input gate.

3. **Candidate Cell State:** The candidate cell state $\tilde{C}_t$ is computed using the current input and the previous hidden state, processed through a tanh activation function:

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \qquad (3)$$

where W_C and b_C are the weight matrix and bias vector for the candidate cell state.

4. **Cell State Update:** The new cell state C_t is updated by combining the previous cell state C_{t-1} and the candidate cell state $\tilde{C}_t$, modulated by the forget gate f_t and the input gate i_t:

$$Ct = ft \odot Ct{-}1 + it \odot \tilde{C}t \qquad (4)$$

where $\odot$ denotes element-wise multiplication.

5. **Output Gate:** The output gate o_t determines the output hidden state h_t, which is used for the next time step's computations and the final output. It is defined as:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \qquad (5)$$

where W_o and b_o are the weight matrix and bias vector for the output gate.

6. **Hidden State:** The hidden state h_t is computed by applying the output gate to the updated cell state:

$$h_t = o_t \odot \tanh(C_t) \qquad (6)$$

The LSTM unit utilizes three gates—forget gate f_t, input gate i_t, and output gate o_t—to regulate the cell state C_t and hidden state h_t. These mechanisms allow LSTMs to retain and manipulate information over long sequences, making them highly effective for tasks involving temporal dependencies. The parameters W_f, W_i, W_C, W_o and b_f, b_i, b_C, b_o are learned during the training process.

LSTMs are particularly effective in capturing long-term dependencies due to their gating mechanisms, making them suitable for various tasks such as language modeling, machine translation, and time series prediction. The ability to maintain and update a cell state helps in preserving relevant information over extended sequences, mitigating issues encountered by conventional RNNs.

Attention-based LSTMs enhance the traditional LSTM architecture by allowing the network to focus on specific parts of the input sequence when predicting the output. This mechanism assigns different levels of importance to different inputs, improving the model's ability to capture relevant features and dependencies. The attention score e_t is calculated as:

$$e_t = \tanh(W_e \cdot h_t + b_e) \qquad (7)$$

The attention weights α_t are obtained using the softmax function:

$$\alpha_t = \frac{\exp(e_t)}{\sum_{t'} \exp(e_{t'})} \qquad (8)$$

The context vector c is a weighted sum of the hidden states:

$$c = \sum_t \alpha_t h_t \qquad (9)$$

Asian Journal of Basic Science & Research
Volume 6, Issue 2, Pages 145-158, April-June 2024

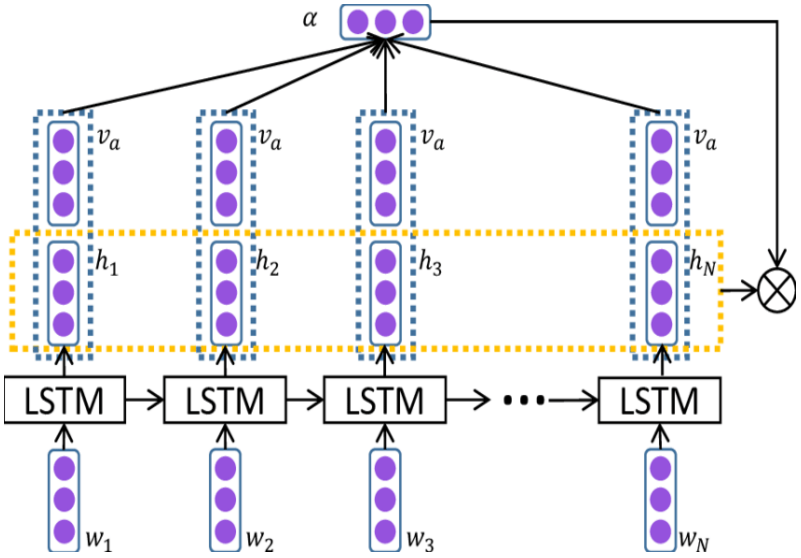

Figure 1. Attention-based LSTM Architecture (Wang et al., 2016)

3.2. XGBoost

XGBoost is a potent tool used in supervised learning tasks, harnessing training data to predict target variables. Fundamentally, XGBoost uses decision trees as its base learners. It operates through an iterative process, adding new base learners in each step to reduce the difference between the predicted values and the actual targets. The final predictions are obtained by summing the outputs of all the individual base learners, resulting in a robust predictive model.

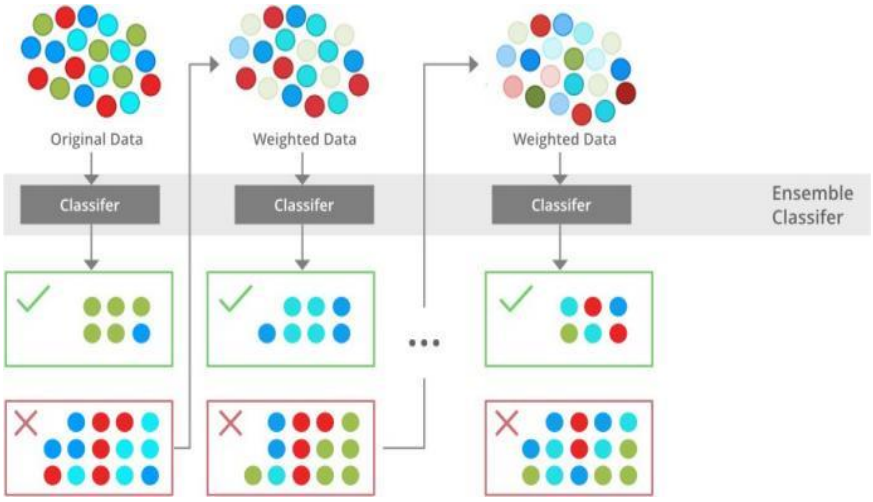

Figure 2. XGBoost Architecture (Verma, 2016)

Conceptually, XGBoost can be viewed as an additive model comprising M decision trees, represented mathematically by below equation (Chen & Guestrin, 2016):

$$Y_i = \sum_{m=1}^{M} f_m(x_i), \quad f_m \in F \qquad (10)$$

where Y_i represents the predicted output or response for the i-th data point, f stands for a decision tree and F represents the function of all decision trees.

During regression, the objective function of the additive model transforms as follows:

$$\mathrm{Obj}(\theta) = \sum_{i=1}^{n} \ell\,(y_i, Y_i) + \sum_{m=1}^{M} \Omega\,(f_m), \quad \theta = (f_1, f_2, \ldots, f_M) \qquad (11)$$

where $\mathrm{Obj}(\theta)$ represents the objective function that XGBoost aims to optimize during the training process, ℓ denotes the loss function and Ω represents the regularization term.

For the regularization terms of each decision tree, we enhance the decision tree using vector mapping. Therefore, the details of $\Omega(f)$ can be seen in below equation:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} \omega_j^2 \qquad (12)$$

where $\Omega(f)$ is the regularization term applied to a single decision tree f within the XGBoost model, T denotes the count of leaf nodes in a decision tree, ω stands for the score vector, while both γ and λ signify the penalty factors.

XGBoost utilizes the forward stage-wise algorithm to simplify model complexity. In each iteration, it adds a decision tree, introducing a new function and its coefficients to more effectively address the residuals from the previous step's predictions. Thus, when the k-th step of learning occurs, the predictive value of x_i is $Y_i^k = Y_i^{k-1} + f_k(x_i)$. Hence, the object function should be expressed by below equation:

$$\begin{aligned} \mathrm{Obj}(\theta)^t &= \sum_{i=1}^{n} \ell\left(y_i, Y_i^t\right) + \sum_{m=1}^{t} \Omega\,(f_m) \\ &= \sum_{i=1}^{n} \ell\left(y_i, Y_i^{k-1} + f_k(x_i)\right) + \sum_{m=1}^{t} \Omega\,(f_m) \end{aligned} \qquad (13)$$

The XGBoost algorithm constructs a comprehensive model by employing the greedy algorithm iteratively in building decision trees. This iterative process involves continually creating decision trees based on the function of the objects involved.

XGBoost presents several advantages that render it highly favored in machine learning. One notable advantage is its exceptional performance and efficiency, particularly evident when dealing with large-scale datasets characterized by high dimensionality. XGBoost's distinctive gradient boosting algorithm optimizes both computational speed and model accuracy. Through techniques such as parallel processing and tree pruning, XGBoost effectively mitigates overfitting while enhancing accuracy, a significant asset in model training. Another notable advantage lies in its flexibility in model tuning, enabling users to adjust complexity and regularization parameters to suit specific use case requirements. Moreover, XGBoost excels in efficiently handling missing data, automatically adapting to address missing values during training, thereby streamlining preprocessing tasks and saving time. Additionally, XGBoost facilitates interpretability through feature importance analysis, empowering users to comprehend the influence of different features on model predictions. These advantages collectively position XGBoost as a robust and adaptable tool for diverse machine learning applications, particularly in domains that prioritize accuracy and efficiency.

3.3. Support Vector Machine

Support Vector Machines (SVM) are a set of supervised learning methods used for classification, regression, and outliers detection. Developed by Vapnik and Cortes (1995), SVMs are particularly powerful for high-dimensional spaces and are effective when the number of dimensions exceeds the number of samples.

In the context of stock data analysis, SVM can be applied to classify the stock's price movement direction or to predict future prices. Given the inherent noise and complexity of financial data, we use Soft Margin SVM, which allows some misclassifications to improve the model's generalization ability.

For non-linearly separable data, such as stock prices, we use Soft Margin SVM. It introduces slack variables $\xi_i \geq 0$ to allow some misclassifications:

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i \qquad (14)$$

The objective function is modified to penalize misclassifications:

$$\min_{w,b,\xi} \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{n} \xi_i \qquad (15)$$

where $C > 0$ is a regularization parameter that controls the trade-off between maximizing the margin and minimizing the classification error.

The optimization problem becomes:

$$\min_{w,b,\xi} \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{n} \xi_i \qquad (16)$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \dots, n \qquad (17)$$

3.4. Hybrid Model

3.4.1. Attention-Based LSTM and XGBoost

XGBoost (Extreme Gradient Boosting) is an ensemble learning method that builds a strong predictive model by combining multiple weak learners, typically decision trees. The objective function of XGBoost combines a loss function and a regularization term:

$$\text{Obj}(\theta) = \sum_{i=1}^{n} L(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \qquad (18)$$

where $\Omega(f)$ is the regularization term that penalizes the complexity of the model:

$$\Omega(f) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2 \qquad (19)$$

Here, T is the number of leaves, w are the leaf weights, and γ and λ are regularization parameters.

In our approach, we use the context vector c from the attention-based LSTM as features for the XGBoost model to predict future stock prices.

3.4.2. Attention-Based LSTM and SVM

Support Vector Machines are supervised learning models used for classification and regression. The SVM constructs a hyperplane in a high-dimensional space to separate different classes. For regression tasks, we use Support Vector Regression (SVR).

The SVR objective function aims to minimize the error within a certain threshold ϵ (epsilon-insensitive loss function):

$$\text{minimize} \quad \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{n} \xi_i + \xi_i^* \qquad (20)$$

subject to:

$$\begin{cases} y_i - (w \cdot x_i + b) \leq \epsilon + \xi_i \\ (w \cdot x_i + b) - y_i \leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases} \qquad (21)$$

where ξ_i and ξ_i^* are slack variables that allow for some flexibility in the margin, and C is a regularization parameter that controls the trade-off between maximizing the margin and minimizing the classification error.

4. Experiment

In this section, we will conduct empirical research using the dataset to evaluate the performance of an attention-based LSTM hybrid model. We will start by providing a basic introduction to the dataset. Following this, we will compare the model's performance with that of other prediction methods.

4.1. Dataset

The dataset used in this research paper comprises historical stock data for Advanced Micro Devices, Inc. (NASDAQ: AMD) spanning from January 1, 2015, to April 30, 2024. This dataset has been sourced from Yahoo Finance, a reputable provider of financial market data. It encompasses a total of 2,347 trading days, offering a comprehensive view of AMD's stock performance over this period.

Each entry in the dataset includes several key variables: Date, Open, High, Low, Close, Volume, and Adjusted Close. The "Date" variable marks the trading day, while "Open" represents the price at which the stock began trading on that day. "High" and "Low" indicate the highest and lowest prices reached during the trading day, respectively. "Close" is the price at which the stock finished trading, and "Volume" reflects the number of shares traded. "Adjusted Close" accounts for any corporate actions such as stock splits or dividends, providing a normalized value for comparison over time.

For example, on January 2, 2015, AMD's stock opened, reached its high, low, and closed at the same price of $2.67, with no trading volume recorded for that day, leading to an adjusted close also at $2.67. This level of detail allows for thorough analysis of the stock's historical performance and instrumental in identifying trends and patterns.

In our experiment, the dataset is divided into two parts. The training dataset, comprising 75% of the observations, is utilized to train our model. The remaining 25% is reserved for testing the model's prediction accuracy. This division ensures that the model is evaluated on unseen data, providing a more robust measure of its performance and generalization capability.

4.2. Evaluation metric

To evaluate the performance of the model, we use the following metrics: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and Coefficient of Determination (R^2).

The Mean Absolute Error (MAE) measures the average magnitude of the errors in a set of predictions, without considering their direction. It is calculated as follows:

$$MAE=\frac{1}{n}\sum_{i=1}^{n}|y_i-\hat{y}_i| \qquad (22)$$

where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the number of observations.

The Root Mean Squared Error (RMSE) is the square root of the average of the squared differences between the predicted and actual values. It is given by:

$$RMSE=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2} \qquad (23)$$

The Mean Absolute Percentage Error (MAPE) expresses accuracy as a percentage of the error. It is defined as:

$$MAPE=\frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i-\hat{y}_i}{y_i}\right|\times 100 \qquad (24)$$

The Coefficient of Determination (R^2) indicates the proportion of the variance in the dependent variable that is predictable from the independent variables. It is calculated as:

$$R^2=1-\frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{\sum_{i=1}^{n}(y_i-\bar{y})^2} \qquad (25)$$

where $\bar{y}$ is the mean of the actual values.

4.3. Result and Discussion

The performance of the various models in predicting stock prices is evaluated using several metrics including Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), and the coefficient of determination (R^2). The LSTM (Long Short-Term Memory) model demonstrates superior performance across most metrics. It achieves the lowest MAE and RMSE values of 7.18 and 9.14 respectively, indicating its ability to make accurate predictions with minimal error. The relatively low MAPE of 0.07 indicates that, on average, the LSTM model's predictions deviate from the actual values by only 7%. Additionally, the high

R^2 value of 0.92 suggests that the LSTM model explains a significant portion of the variance in the observed data, indicating a strong fit.

Table 1. Comparison of Different Models

Model	MAE	RMSE	MAPE	R^2
LSTM	7.18	9.14	0.07	0.92
XGBoost	15.04	27.02	0.11	0.33
SVM	15.61	29.33	0.11	0.21
LSTM+XGBoost	8.04	13.93	0.06	0.82
LSTM+SVM	14.09	21.86	0.11	0.56

In contrast, the XGBoost and SVM (Support Vector Machine) models perform less favorably. Both models exhibit higher MAE, RMSE, and MAPE values compared to the LSTM model, indicating larger prediction errors and less accuracy. The relatively low R^2 values for XGBoost (0.33) and SVM (0.21) suggest that these models explain a smaller proportion of the variance in the data compared to the LSTM model.

The combined models, LSTM+XGBoost and LSTM+SVM, show mixed results. While LSTM+XGBoost performs reasonably well with lower MAE, RMSE, and MAPE compared to individual XGBoost and SVM models, its performance is still inferior to the standalone LSTM model. Similarly, LSTM+SVM exhibits higher errors and lower R^2 compared to the standalone LSTM model, indicating that the addition of SVM does not significantly improve predictive performance.

Overall, these results suggest that the LSTM model outperforms both traditional machine learning algorithms (XGBoost and SVM) and combined models in predicting stock prices. Its ability to capture temporal dependencies and long-range patterns in sequential data makes it a robust choice for financial time series forecasting.

5. Conclusion

In this study, we explored the efficacy of different models, including LSTM, XGBoost, SVM, and hybrid models combining LSTM with XGBoost and SVM, for stock price prediction. The results, as shown in Table 1, indicate that the standalone LSTM model outperformed the other models with an MAE of 7.18, an RMSE of 9.14, a MAPE of 0.07, and an R2 value of 0.92. Among the hybrid models, the LSTM+XGBoost model demonstrated superior performance compared to LSTM+SVM, with an MAE of 8.04, an RMSE of 13.93, a MAPE of 0.06, and an R2 value of 0.82. This suggests that while hybrid models can enhance predictive performance, the combination of LSTM with XGBoost is more effective than with SVM in this context. Despite these promising results, there is significant room for further improvement. Enhanced feature engineering could incorporate additional features such as macroeconomic indicators, market sentiment data, and technical analysis indicators, thereby providing a more comprehensive representation of the factors influencing stock prices. Advanced hybrid models, such as integrating LSTM with other machine learning algorithms like Random Forest or CatBoost, could also be explored to potentially yield better results.

Attention mechanisms within the LSTM framework could be further refined to dynamically prioritize more relevant information, improving the model's ability to handle complex and volatile financial data. Additionally, enhancing model interpretability is crucial, particularly in financial applications where understanding the decision-making process is important. Techniques such as SHapley Additive exPlanations (SHAP) or Local Interpretable Model-agnostic Explanations (LIME) could be employed to achieve this.

Data augmentation techniques could be leveraged to artificially increase the size of the training dataset, addressing the issue of limited data and improving model generalization. Moreover, implementing real-time prediction capabilities would allow for immediate reaction to market changes, enhancing the practical applicability of the models in live trading scenarios. In conclusion, while LSTM and its hybrid models with XGBoost and SVM show substantial promise in stock price prediction, there remains considerable potential for enhancement through advanced techniques and additional research efforts.

Future studies could delve into the development of more sophisticated hybrid models by combining LSTM with other advanced machine learning algorithms such as Random Forest or CatBoost, which could potentially yield superior results. Another avenue for improvement involves enhanced feature engineering, where incorporating additional features like macroeconomic indicators, market sentiment data, and technical analysis indicators could provide a more comprehensive representation of the factors influencing stock prices. Furthermore, refining attention mechanisms within the LSTM framework to dynamically prioritize more relevant information could significantly improve the model's ability to handle complex and volatile financial data. Enhancing model interpretability is also crucial; implementing techniques such as SHapley Additive exPlanations (SHAP) or Local Interpretable Model-agnostic Explanations (LIME) can help demystify the decision-making process, which is particularly important in financial applications. Finally, employing data augmentation techniques to artificially increase the size of the training dataset can address the issue of limited data and improve model generalization, while integrating real-time prediction capabilities would allow for immediate reactions to market changes, enhancing the practical applicability of these models in live trading scenarios. By addressing these areas, future studies can further improve the accuracy and reliability of financial forecasting models, contributing to more effective and informed decision-making in the financial markets.

Declarations

Source of Funding

This study did not receive any grant from funding agencies in the public, commercial, or not-for-profit sectors.

Competing Interests Statement

The author declares having no competing interest with any party concerned during this publication.

Consent for Publication

The author declares that he/she consented to the publication of this study.

Authors' contributions

All research work is from Jiawei Zhou.

Availability of data and material

All data pertaining to the research is kept in good custody by the author.

References

Hu, Y., Feng, B., Zhang, X., Ngai, E.W.T., & Liu, M. (2015). Stock trading rule discovery with an evolutionary trend following model. Expert Syst. Appl., 42(1): 212–222. https://doi.org/10.1016/j.eswa.2014.07.040.

Asteriou, D., & Hall, S.G. (2011). ARIMA models and the Box-Jenkins methodology. Appl. Econom., 32(3): 265–286. https://doi.org/10.1080/00036840802600071.

Engle, R.F. (2001). GARCH 101: The use of ARCH/GARCH models in applied econometrics. J. Econ. Perspect., 15(4): 157–168. https://doi.org/10.1257/jep.15.4.157.

Jin, C., Shi, Z., Zhang, H., & Yin, Y. (2021). Predicting lncRNA–Protein Interactions Based on Graph Autoencoders and Collaborative Training. Int. Conf. Bioinform. Biomed. (BIBM), IEEE. https://doi.org/10.1109/bibm52615.2021.9669496.

Srivastava, N. (2013). Improving neural networks with dropout. Univ. Toronto, 182(566): 7. https://doi.org/10.1201/b15320.

Khandelwal, I., Satija, U., & Adhikari, R. (2015). Forecasting seasonal time series with functional link artificial neural network. Int. Conf. Signal Process. Integr. Networks, IEEE, Pages 725–729. https://doi.org/10.1109/spin.2015.7095311.

Jin, C., Shi, Z., Li, W., & Guo, Y. (2021). Bidirectional LSTM-CRF Attention-Based Model for Chinese Word Segmentation. https://doi.org/10.1109/bibm52615.2021.9669425.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., et al. (2017). Attention Is All You Need. NeurIPS, Pages 6000–6010. https://doi.org/10.48550/arxiv.1706.03762.

Moritz, B., & Zimmermann, T. (2016). Tree-based conditional portfolio sorts: The relation between past and future stock returns. Available at SSRN 2740751.

Takeuchi, L., & Lee, Y.Y.A. (2013). Applying deep learning to enhance momentum trading strategies in stocks. In Technical Report, Stanford University.

Allen, F., & Karjalainen, R. (1999). Using genetic algorithms to find technical trading rules. Journal of Financial Economics, 51(2): 245–271. https://doi.org/10.1016/s0304-405x(98)00052-x.

Kurani, A., Doshi, P., Vakharia, A., & Shah, M. (2023). A comprehensive comparative study of Artificial Neural Network (ANN) and Support Vector Machines (SVM) on stock forecasting. Annals of Data Science, 10(1): 183–208.

Batres-Estrada, B. (2015). Deep learning for multivariate financial time series (Dissertation). Retrieved from https://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-168751.

Jing, N.W.Z., & Wang, H. (2021). A hybrid model integrating deep learning with investor sentiment analysis for stock price prediction. Expert Systems with Applications, 178: 115019. https://doi.org/10.1016/j.eswa.2021.115019.

Aggarwal, P.K., Mittal, R., Ganaie, A., & Ishfaq, M. (2023). Stock prediction by integrating sentiment scores of financial news and MLP-Regressor: A machine learning approach. Procedia Computer Science, 218: 1067–1078. https://doi.org/10.1016/j.procs.2023.01.086.

Sen, J., Mehtab, S., & Dutta, A. (2021). Stock price prediction using machine learning and LSTM-based deep learning models. https://doi.org/10.36227/techrxiv.15103602.

Wu, J.M.T., Li, Z., Herencsar, N., Vo, B., & Lin, J. (2021). A graph-based CNN-LSTM stock price prediction algorithm with leading indicators. Multimedia Systems, 29. https://doi.org/10.1007/s00530-021-00758-w.

Zhang, Z. (2022). Research on stock price prediction based on PCA-LSTM model. Academic Journal of Business Management, 4(3): 42–47. https://doi.org/10.25236/AJBM.2022.040308.

Gers, F.A., Schmidhuber, J., & Cummins, F.A. (2000). Learning to forget: Continual prediction with LSTM. Neural Computation, 12(10): 2451–2471.

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Pages 785–794. https://doi.org/10.1145/2939672.2939785.

Verma, N. (2022). XGBoost algorithm explained in less than 5 minutes. Medium. https://medium.com/@techynilesh/xgboost-algorithm-explained-in-less-than-5-minutes-b561dcc1ccee.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20: 273–297.

Wang, Y., Huang, M., Zhu, X., & Zhao, L. (2016). Attention-based LSTM for aspect-level sentiment classification. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Pages 606–615. https://doi.org/10.18653/v1/D16-1058.